Acoustic Scene Recognition Using Transfer Learning

Nidhi Sethi

Submitted for the Degree of Master of Science in

MSc Artificial Intelligence

Department of Computer Science
Royal Holloway University of London
Egham, Surrey TW20 0EX, UK

7th September 2022

Declaration

This report has been adapted on the basis of my own work. Where other published and unpublished source materials have been used, these have been acknowledged.

Word Count: 6572

Student Name: Nidhi Sethi

Date of Submission: 7th September

Signature: Nidhi Sethi

Abstract

Background: Sound complements visual input and is an important modality for perceiving the environment. Machines in various environments are increasingly equipped with the ability to hear, including smartphones, autonomous robots, and security systems. In this work, we handle state-of-the-art deep learning models that have revolutionized speech recognition to understand common environmental noises.

Aim: This work aims to classify ten different acoustic scenes using environmental sounds. We use spectrograms as visual inputs and feed them convolutional neural networks to make predictions.

Data: We use a dataset from the Detection and Classification of Acoustic Scenes and Events (DCASE). The dataset contains ten acoustic scenes, such as airports, parks, trams, metro stations, buses and etc., totalling 240 minutes of audio recording.

Methods: A log-amplitude Mel-spectrogram is used by our system as the model input. Each audio file is first converted to mono and loaded with a sampling rate of 48 kHz. The audio data is then transformed into a spectrogram using a short-time Fourier transformation, which is produced using an 8192-point FFT. We then augment data and apply the VGG16 pre-trained model combined with two layered classifiers.

Results: Our experiments show that model performance doesn't increase as we increase CNN layers and the use of a pre-trained model CNN is performing better rather than a conventional CNN. We also experimented with the VGG19 model, but the results were almost similar, so to reduce complexity, we experimented more with VGG16. We used the TAU Urban Acoustic Scenes 2019 development dataset for training and cross-validation, resulting in a more than 10% improvement when compared to the baseline system.

Conclusions. We found that pre-trained models can decrease complexity with better results. Audio data, when converted to Mel-spectrograms, tends to suppress low frequencies, so log frequency spectrograms were used for appropriate analysis. Data augmentation is played a key role, and with proper control of parameters, we can predict better. With regular time and frequency masking, not only do we avoid overfitting we also achieve far better results. We also discovered that the most influential features of spectrograms after data augmentation depended on frequency masking width and time size window.

Contents

1. INTRODUCTION

Sound in our surroundings gives us context about our location and the adjacent physical activities. Humans are able to recognize their environment or contexts (such as a park, beach, or bus) with simply aural information. For systems that try to automate it, however, this process is not simple. One of the objectives of machine listening, which is attracting increasing attention from the scientific community, is to create systems that perform similarly to people in tasks like this. Images and videos were once thought of as a way to identify places, but new approaches were required because of concerns about individual privacy. With some investigation and arithmetic skills, spectrograms opened new opportunities.

To be more specific, there is growing interest in the auditory study of urban environments, which has been made possible in part by multimedia sensor networks and a wealth of online multimedia information featuring urban settings. However, while there is a substantial corpus of study in fields connected to acoustics, such as speech, music, and bioacoustics, there is comparatively little work done on analyzing urban acoustic settings.

Innovative ideas that have the potential to provide novel solutions to challenging and complex challenges, such as personalized location tagging with respect to noise levels/types, etc., as well as to generic concerns like noise-pollution control and reduction, are given significant study exposure. The task of approval lAbel to an audio stream in order to identify the context or area in which the audio stream was captured, such as a park or beach, is known as acoustic scene classification, or ASC.

ASC systems in the past relied on a feature engineering method, in which the audio signal was processed to extract pre-built low-level features, which were then fed into a classifier. Cepstral features, such as MFCCs, are the most often used hand-crafted features in audio-related activities. c and are also one of the most common in ASC. GMM and SVM are typical examples of classifiers used in ASC. The feature engineering approach strongly relies on the pre-designed features' ability to extract pertinent information from the signal, which may require a lot of skill and work. Given the vast range of issues and unique examples found in the numerous fields, this strategy really proved to be neither effective nor sustainable.

Similar to paradigm shifts seen in fields like computer vision or speech recognition, ASC has recently undergone one. Based on deriving representations from data, new techniques have emerged. Significant research advancements have been made possible by these data-driven methodologies, particularly deep learning, which have spread quickly throughout the audio research community. In this situation, feature extraction and classification are jointly optimized as the system is able to learn internal representations from a simpler one at the input (usually a time-frequency representation). Convolutional Neural Networks (CNNs) have emerged as one of the most successful deep learning techniques for a number of audio-related tasks, including speech recognition, automatic music labelling, and environmental sound classification. Particularly for ASC, CNNs have also been successfully used.

The goal of this research is to define the job of acoustic scene identification, which chooses a semantic label to describe the acoustic surroundings of an audio stream. Existing work for this task largely uses conventional classifiers such as GMMs and SVMs, which do not have

the feature abstraction found in deeper models. Furthermore, traditional models rely on feature extraction processes to capture local temporal dynamics rather than modelling temporal dynamics themselves. For instance, the winning answer proposed by Roma et al. (2013) for one of the IEEE challenge on Detection and allocation of Acoustic Scenes and Events (DCASE) challenge extracts MFCC and temporal features using Recurrence Quantification Analysis over a short time window.

In this report, we have proposed a CNN model using transfer learning evaluated by the classifier, which is trained with various training settings. We are experimenting with two different pre-trained modes VGG16 and ResNet50, both compared with seven layered CNN baseline model [1], which is described further in the later part of the report. To avoid overfitting, we use two data augmentation methods time and frequency masking. After training, we discover the two most influential parameters for the model which yield the best accuracy. The following report will cover a detailed explanation of system architecture, experiments, experimental results, and a conclusion.

2. RELATED WORK

Scene classification is commonly studied in both audio and video domains. For acoustic scene classification, the input is typically a short audio recording, while for visual scene classification tasks, the input can be an image or a short video clip. ASC's state-of-the-art solutions are based on spectral features (most commonly logmel spectrograms) and convolutional neural network architectures commonly used in large ensembles. In comparison, visual scene classification (VSC) from images has a long history and a larger variety of approaches. Global attribute descriptors, spatial layout pattern learning, discriminative region detection, and most recently, deep hybrid models. As large image datasets such as ImageNet became available, image classification performance improved significantly. In recent years, various network structures have been studied. For CNN's, transfer learning has recently been proposed to further improve performance.

ASC is currently attracting the attention of many scientists in the field. Many researchers have proposed various acoustic scene classification systems to identify these scenes. Many features and methods are currently used to classify acoustic scenes. Acoustic properties of interest include Mel- frequency cepstral coefficients, Mel spectrograms, constant-Q transform spectrograms, and other low-level properties. Common methods include Support Vector Machines (SVM), Hidden Markov Models (HMM), and Gaussian Mixture Models (GMM). With the continuous development of deep learning technology, neural networks are quickly becoming the mainstream solution for ASC. In particular, we have successfully used convolutional neural networks (CNNs) to classify acoustic scenes, achieving state-of-the-art performance. Transfer learning works as a game changer. By combining different pre-trained models, researchers have achieved 85%. Studies have shown using data augmentation methods like time, and frequency masking can help increase accuracy.

The IEEE AASP Challenge on Detection and Classification of Acoustic Scenes and Events (DCASE) attracted many researchers from academia and industry [2-5].

From previous DCASE challenge results, deep learning, such as CNN [6], CRNN [7] and ensemble learning [8], shows great performance in this area. The audio data is mostly generated the time-frequency info, such as Mel Frequency Cepstral Coefficients (MFCC) [9], Constant Q transformation (QCT) [10], and other audio signal features [10]. From last year's top rank results, the CNN network achieved good performance, RNN network shows better performance in time correlation signal.

A paper submitted in DCASE 2016 said Transfer learning could be used for enhancing the detection accuracy of real-life sound events that were investigated using a synthetic source database. An improvement in acoustic event detection was noticed with transfer learning using the CRNN approach if a substantial amount of source data, which is representative of a diverse set of events.[11]

We are using the investigation [11] to propose a model using transfer learning with a larger dataset (DCASE 2019). In the baseline system with no data augmentation, the model achieved 62% accuracy on the development dataset. As suggested in many papers [11], a diverse dataset can produce better results, so we apply data augmentation techniques to spectrograms. Using augmented data and a simple transfer learning model we were able to achieve 71% accuracy.

3. SYSTEM ARCHITECTURE

The overall flow diagram of our proposed system is illustrated in Figure 1. For the extraction, the audio samples are processed to extract features. Subsequently, for each audio recording, a Mel spectrogram is extracted. Ambient sounds like in a bus or park can have a lot of low-frequency sounds which can sometimes be ignored by mel scale. To overcome this challenge and improve analysis, we extract log-frequency spectrograms. In the training stage, we use black and white spectrograms as input and use the spec augment [13] technique for data augmentation. The CNN-based classifier will be trained using the extracted features from all training data using the transfer learning process, which will be covered in more detail in the following section. In the testing stage, we use the best model from the training stage and feed random data from the dataset of DCASE 2019 [14], where there is data from two different cities which is not known to the model. The same data pre-processing is used on the new data and fed to the saved model from the training phase. Finally, we use the model to find the best set of parameters to get the best accuracy.

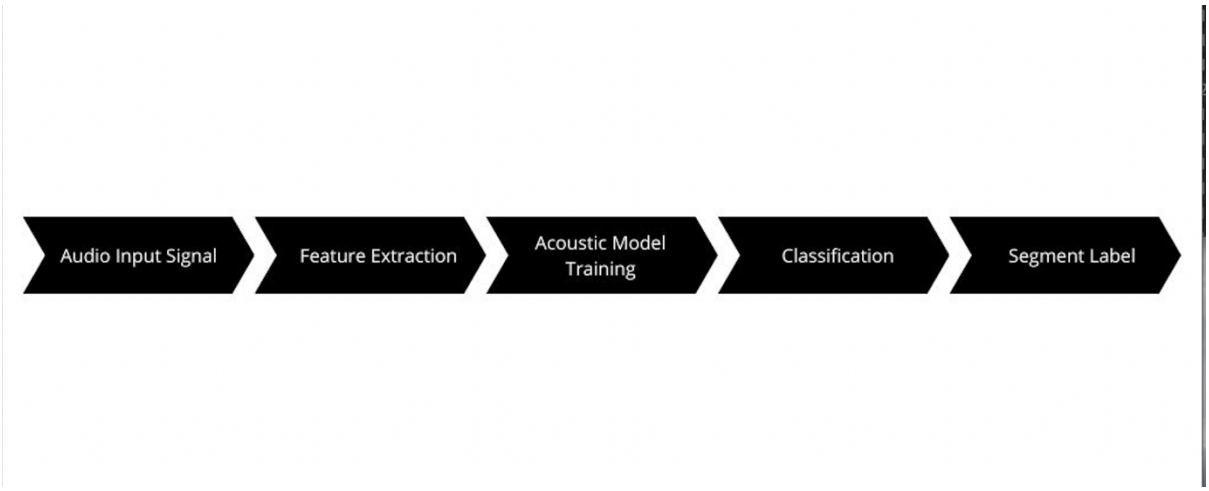


Figure 1: General schematic diagram for the proposed system

3.1 Data and Dataset

The TAU Urban Acoustic Scenes 2019 development dataset is used for Task 1A[31] of the DCASE 2019 Challenge. The dataset consists of audios from 10 different acoustic scenes and each segment is 10 seconds long. Each acoustic scene has 1440 segments (240 minutes of audio) recorded with the same device. The acoustic scenes include the airport, shopping mall, metro station, pedestrian street, public square, traffic street, tram, bus, metro, and park. For each scene class, the recording takes place at a different location; for each recording position, there are 5-6 minutes of audio, and the original recording is divided into segments of 10 seconds in length. The methodology's main strategy was to turn audio files (.wav) into spectrograms (.png). Mel Spectrograms were discovered to be the ideal agent for representing the many aspects of an audio file, which is also highly important for data analysis and research.

3.2 Spectrograms

Audio files can be converted into spectrograms. A spectrogram is a simple "snapshot" of a sound wave, and because it is an image, it makes a good input for CNN-based architectures designed to process images. A spectrogram is generated from the audio signal using the Fourier transform. The Fourier transform decomposes the signal into individual frequencies and displays the amplitude of each frequency present in the signal. A spectrogram divides the duration of an audio signal into short time segments and applies a Fourier transform to each segment to determine the frequencies contained in that segment. Then combine the Fourier transforms of all these segments into one graph. Plot frequency (y-axis) versus time (x-axis) and use different colours to indicate the amplitude of each frequency. The brighter the colour, the higher the energy of the signal. If the spectrogram is used as it is, high-dimensional arrays and processing become difficult, so we developed the model-spectrogram using the mel-scale. Displays colour on a decibel scale instead of the Y-axis frequency and amplitude. It uses the STFT (Short-Time Fourier Transform) to represent audio signals and improve data sampling more accurately.[24]

The Mel scale is predicated on what humans perceive as equal pitch differences. The Mel scale defines how the frequency axis is scaled within the spectrograms.

1. Evidently, the results of the scaling produced thanks to using of Mel Spectrograms densely distributed the low frequencies, but at the same time, it almost filtered out most of the high-frequency tones. In technical language, we will say that the information represented was a touch skewed towards higher frequencies.

2. To delineate this issue, the scaled spectrograms were further knowledgeable of a logarithmic re-scaler, here within the sort of Log-Frequency Power Spectrogram, also called constant-Q power spectrogram. With the utilization of log-frequency spectrograms, the structure of low frequencies gets well represented. Also, the audio power is now more distributed over the frequency intervals.

3. Yet high frequencies were still underrepresented. To delineate this issue, an emphasizing algorithm was used, which gradually toned the upper frequencies and glued their representation. Broadly, the emphasizing algorithm filter mixes the signal with its derivative. Mathematically,

$$y[n]= x[n]-(\alpha \cdot x[n-1]) / (1-\alpha)$$

Equation 1: Pre-emphasis where α is 0.97

The above equation is summarised formula of the algorithm, where α may be a constant (signal modulator). The code for the identical is employed in Pre-Emphasis a part of the environment.

4. As a final block within the module, the resultant spectrograms are now converted to grey-scale images. This accounts for the simplification of the analysis complexity. These images act as input for the model, which shall be discussed further.

3.3 Feature extraction and normalization

The performance of the acoustic scene classification has upper bounds that are determined by the effectiveness of the features, and the classifier decides the degree to which performance approaches the upper limit. As a result, feature extraction is crucial for the classification of acoustic scenes in audio analysis. Feature extraction can be used in the audio analysis system to convert the signal into a representation. With little memory and processing power needed, it may compactly and redundantly represent the audio.

In general, it can be challenging to directly interpret the time domain characteristics of a sound signal. The majority of the time, domain elements make it practically impossible to distinguish across sound scenes. Therefore, frequency-domain features and time-frequency domain features have been used to represent sound signals that are more in line with human perception.

A log-amplitude Mel-spectrogram is used by our system as the model input. Each audio file is first converted to mono and loaded with a sampling rate of 48 kHz. The audio data is then transformed into a spectrogram using a short-time Fourier transformation, which is produced using an 8192-point FFT on a 40-millisecond Hamming window with a 20-millisecond overlap. The spectrogram is turned into a log-scale and 256 bands of mel-scaled features. The size of the input feature finally obtained is 500 frames in time and 256 bands in frequency. All features are mean-centred and normalized along with the individual frequency bins.

3.4 Data augmentation

TAU Urban Acoustic Scenes 2019 development dataset contains only data comes from 10 of the 12 cities. Therefore, overfitting the development dataset will result in poor classification accuracy. To overcome this overfitting problem, we have used a method called Spec Augment to mix some noise into training data. Spec Augment is a data augmentation technique that can be applied in the spectrogram domain and performs augmentation by applying three methods: time masking, frequency masking, and time stretching. With the exception of the time stretching technique, we have created additional training data using two-time masks and two frequency masks per data. However, the experiment produced poor classification accuracy, suggesting that this added too much noise to the training data. As a result, we modified to only apply the Spec Augment on the data that are randomly chosen for each stage of model learning. In this situation, there might be a class imbalance in the training sets that are generated repeatedly, but the ensemble method can fix this.

We use two types of data augmentation in our experiment. After pre-emphasis on the log frequency or constant Q power spectrum (figure 2), we apply our data augmentation. In the random audio transform method, we transform spectrograms randomly in three ways here using hyperparameters image size, number of octaves by which spectrograms will shift, number of pixels per octave, increment in tempo and sample size. Firstly, spectrograms frequencies are transposed randomly, then the tempo is changed randomly, and lastly, a random sampling window in time is chosen for transformation. Then we apply frequency masking, in which we randomly mask a frequency band of various widths. We can change the width of the band of masked frequency and alter the side. Figure (3) shows random horizontal frequency masking of 0.2 widths of the grey spectrogram.

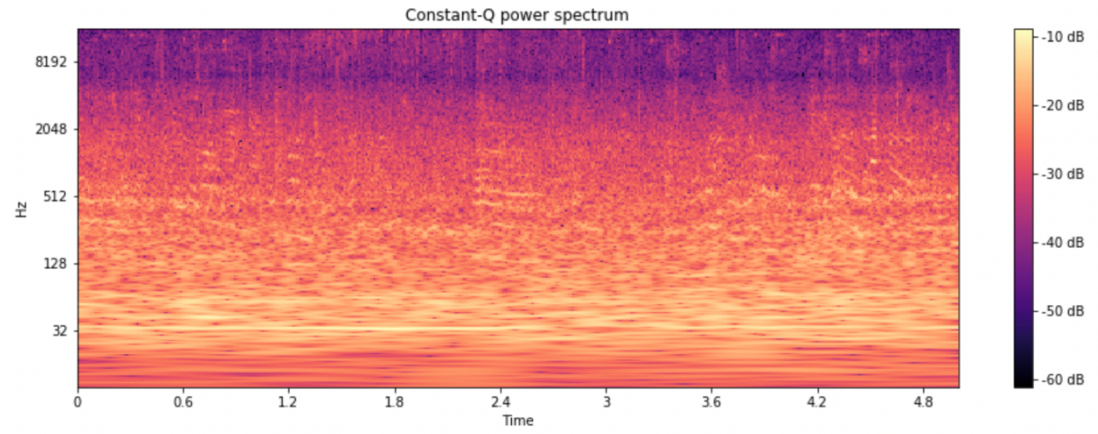


Figure 2: Log frequency Spectrogram

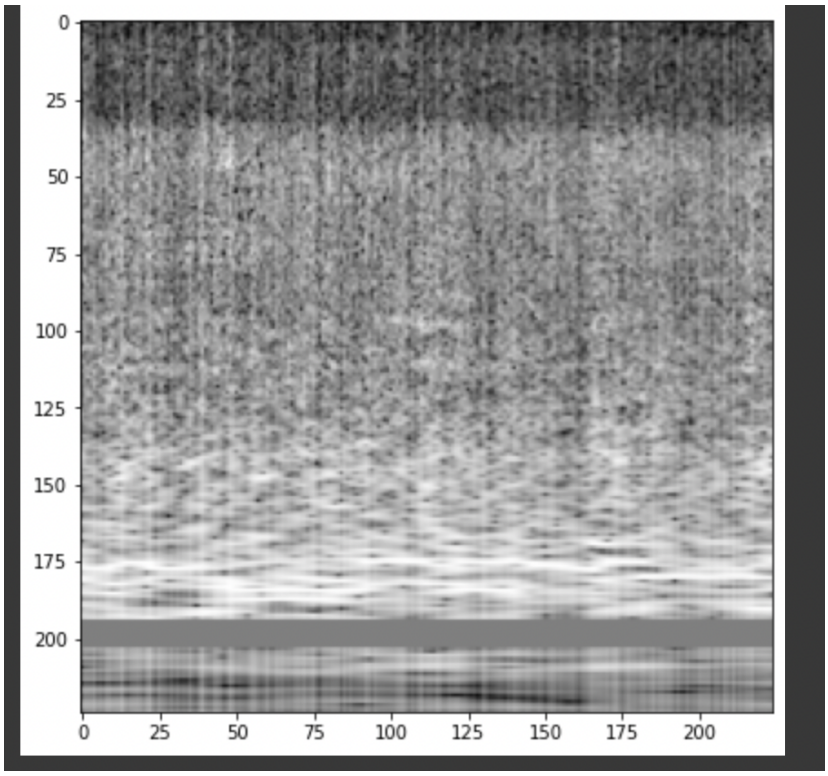


Figure 3: Spectrogram (Final input to model)

3.5 CNN-based Transfer Learning

Recent advances in computer vision and image clarification have been made possible by convolutional neural networks (CNNs). CNNs automatically identify the characteristics required for the classification of images based only on the raw pixel intensity data. Since segmentation and classification are combined in a single framework, picture classification may be done without first performing cell segmentation, and this supervised feature learning technique has been demonstrated to be superior to utilizing traditional handmade features. The pipeline workflow of the conventional methods is handled by the network itself, which is a useful feature of CNN. Additionally, local connectivity and parameter sharing maintain the number of parameters quite low, even for a deeper network, by applying convolving filters on the input layers.

The early CNN layers capture low-level features like edges and blobs—properties that are frequently shared between various types of images—which contributes to the usefulness of transfer learning. Transfer learning, like unsupervised pretraining, may also contribute to testing set error reduction when there are few labelled examples by imposing a regularising effect. The parameters were moved into an area of parameter space that, while better, is not accessible based just on the labelled data given, which is another aspect that is likely to be to blame for the improved performance.[22]

CNN has been used for acoustic scene classification [16, 17]. CNN can automatically learn the feature representation as opposed to conventional handmade feature extraction. Additionally, it has been demonstrated that on the job of classifying acoustic scenes, CNN-based systems frequently outperform conventional machine learning techniques [15]. The basic components of CNN are convolution, pooling, and activation layers. The basic components of CNN are convolution, pooling, and activation layers. The typical CNNs, including AlexNet [18]. Oxford VGGNet [19], ResNet [20] take fixed-size input. By simply eliminating the AlexNet/VGGNet/ResNet's fully connected layers, we convert it into a fully convolutional network (FCN). A softmax layer is added to these networks with ten nodes acting as each of the 10 scene classes in order to use them for acoustic scene classification. AlexNet/VGGNet/ResNet had remarkable success in the ImageNet [18] picture categorization assignment. In 2012, AlexNet emerged victorious from the ILSVRC (ImageNet Large Scale Visual Recognition Competition). The 2014 ILSVRC competition's victor was VGG. Additionally, ResNet recently took first place in several competitions in the field of computer vision. This is the reason that, in the experiment, we compare these CNN topologies. We should take into account the input and output adaption issues while applying transfer learning from image classification to audio scene classification. Different from image classification tasks often containing RGB three channels, we convert spectrograms to black and white as the input. As image inputs are spectrograms, augmentation methods like rotating or flipping don't work. Therefore, we use masking and shifting. We train a fully convolutional network with a pre-trained model with augmented data and add a custom classifier in the end to get a proper classification.

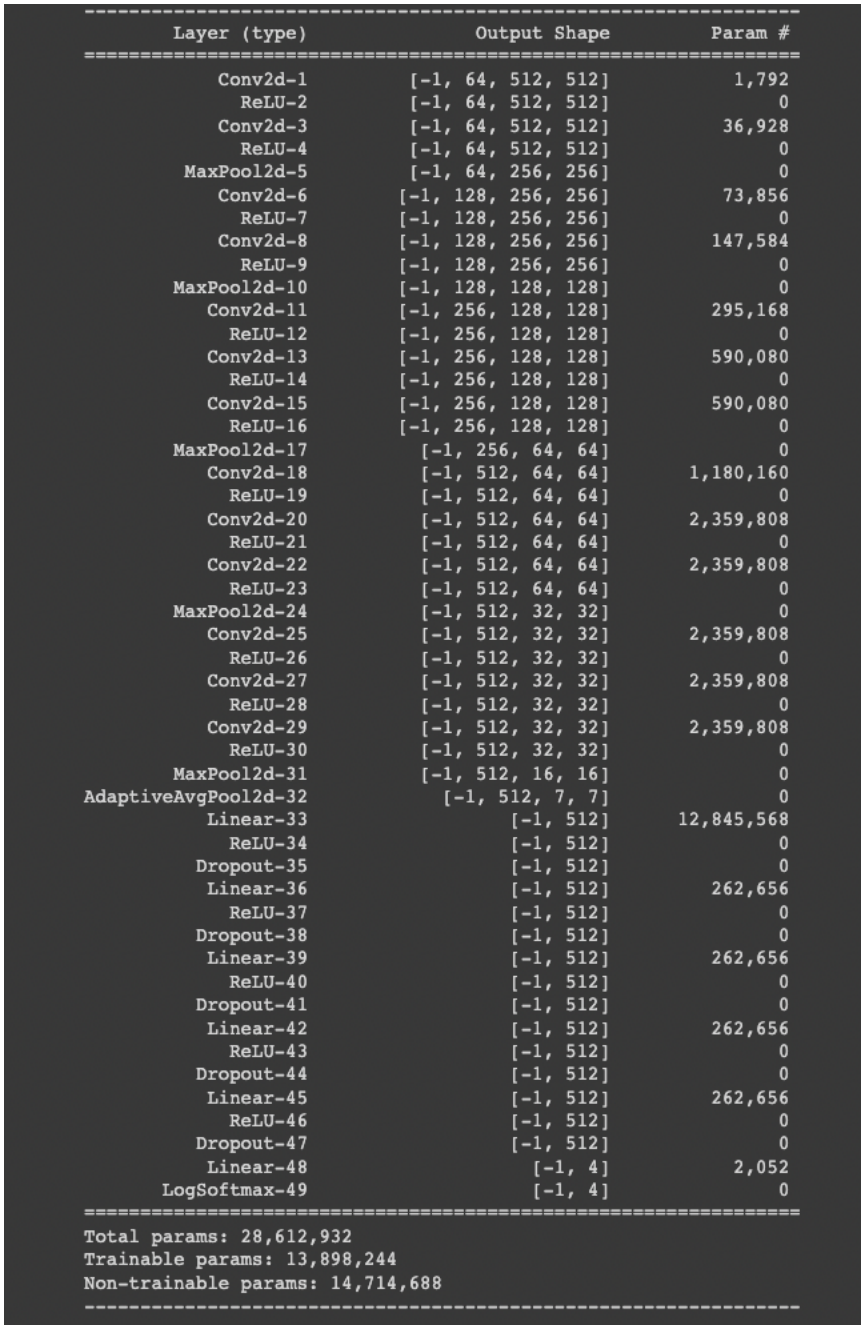
3.6 VGG16 Architecture

CNN has achieved state-of-the-art performance in image classification. A CNN consists of several convolutional layers followed by fully connected layers. Each convolutional layer consists of filters to convolve with the output from the previous convolutional layers. The filters can capture local patterns in feature maps, such as edges in lowers layers and complex profiles in higher layers. Meanwhile, CNNs have been applied to many audios' classification and sound event detection tasks using inputs such as log Mel-spectrogram.
VGGNet model is not fundamentally different from the traditional CNN models in principle and has been gradually applied to acoustic scene classification- Net [9] uses a 3*3 small convolution core to improve performance by deepening the network structure.
The VGG network was proposed to decompose the 5*5 kernel to a convolutional network block consisting of two cascaded convolutional layers with 3*3 kernels. So, we applied the 13-layer CNN and then connected it with six linear layers. The ReLU (Rectified Linear Unit) function is used as a non-linearity after every convolutional layer. Average pooling or max pooling with a size of 2*2 is applied after each convolutional block to reduce the feature map size. After the adaptive average pooling layers further reduced size, we created our own custom classifier containing linear layers. In the neural network's final phases, the linear layer is employed. This layer aids in altering the output's dimensionality from the layer before so that the model may more easily establish the relationship between the values of the data it is working with. The input characteristics of linear layers are multiplied by a weight matrix to create the output features. The input characteristics are sent to a linear layer as a one-dimensional tensor that has been flattened, and they are then multiplied by the weight matrix. Every linear layer is followed by ReLU and a dropout layer to avoid overfitting. The last layer is the activation function classified by the log SoftMax function. A SoftMax function's logarithm is what the log SoftMax function is. Using log probabilities entails using a logarithmic scale to depict probabilities rather than the conventional [0,1] range. Because of the way computers work, using log probabilities increases numerical stability when the probabilities are very tiny. In many cases, taking the product of a large number of probabilities in log form is quicker.

$$\text{LogSoftmax}(x_i) = \log\left(\frac{\exp(x_i)}{\sum_j \exp(x_j)}\right) \quad [25]$$

Equation 2: log of the softmax function

The VGG-net architecture is depicted in figure 4. The loss function is the negative log-likelihood loss (NLLLoss), and the optimizer is ADAM. ADAM proved to be the best optimizer when experimented with others. Also, we don't need to focus so much on the learning rate as it is said to make adjustments to the learning rate in the training phase by reducing the learning rate to a pre-defined schedule [28]. Also, with the log SoftMax activation function, NLLLoss works better it performs the masking step by mean reduction. Also, our classification problem outputs are dimensional so NLLLoss seems a better option than cross-entropy loss.[29]. The system is implemented using the PyTorch library in python language and adapted from GitHub references[32].



Layer (type)	Output Shape	Param #
Conv2d-1	[-1, 64, 512, 512]	1,792
ReLU-2	[-1, 64, 512, 512]	0
Conv2d-3	[-1, 64, 512, 512]	36,928
ReLU-4	[-1, 64, 512, 512]	0
MaxPool2d-5	[-1, 64, 256, 256]	0
Conv2d-6	[-1, 128, 256, 256]	73,856
ReLU-7	[-1, 128, 256, 256]	0
Conv2d-8	[-1, 128, 256, 256]	147,584
ReLU-9	[-1, 128, 256, 256]	0
MaxPool2d-10	[-1, 128, 128, 128]	0
Conv2d-11	[-1, 256, 128, 128]	295,168
ReLU-12	[-1, 256, 128, 128]	0
Conv2d-13	[-1, 256, 128, 128]	590,080
ReLU-14	[-1, 256, 128, 128]	0
Conv2d-15	[-1, 256, 128, 128]	590,080
ReLU-16	[-1, 256, 128, 128]	0
MaxPool2d-17	[-1, 256, 64, 64]	0
Conv2d-18	[-1, 512, 64, 64]	1,180,160
ReLU-19	[-1, 512, 64, 64]	0
Conv2d-20	[-1, 512, 64, 64]	2,359,808
ReLU-21	[-1, 512, 64, 64]	0
Conv2d-22	[-1, 512, 64, 64]	2,359,808
ReLU-23	[-1, 512, 64, 64]	0
MaxPool2d-24	[-1, 512, 32, 32]	0
Conv2d-25	[-1, 512, 32, 32]	2,359,808
ReLU-26	[-1, 512, 32, 32]	0
Conv2d-27	[-1, 512, 32, 32]	2,359,808
ReLU-28	[-1, 512, 32, 32]	0
Conv2d-29	[-1, 512, 32, 32]	2,359,808
ReLU-30	[-1, 512, 32, 32]	0
MaxPool2d-31	[-1, 512, 16, 16]	0
AdaptiveAvgPool2d-32	[-1, 512, 7, 7]	0
Linear-33	[-1, 512]	12,845,568
ReLU-34	[-1, 512]	0
Dropout-35	[-1, 512]	0
Linear-36	[-1, 512]	262,656
ReLU-37	[-1, 512]	0
Dropout-38	[-1, 512]	0
Linear-39	[-1, 512]	262,656
ReLU-40	[-1, 512]	0
Dropout-41	[-1, 512]	0
Linear-42	[-1, 512]	262,656
ReLU-43	[-1, 512]	0
Dropout-44	[-1, 512]	0
Linear-45	[-1, 512]	262,656
ReLU-46	[-1, 512]	0
Dropout-47	[-1, 512]	0
Linear-48	[-1, 4]	2,052
LogSoftmax-49	[-1, 4]	0

Total params: 28,612,932
Trainable params: 13,898,244
Non-trainable params: 14,714,688

Figure 4: Model Summary

3.7 Transforms

Data Augmentation is extremely important to regularize your network and increase the dimensions of your training set. There are many Data transformation techniques like rotation, flip, crop, etc., which will change the images' pixel values but still keep almost the full information of the image, in order that humans could hardly tell whether it had been augmented or not. This forces the model to be more flexible with the big variation of objects inside the image, regarding less on position, orientation, size, and colour. Models trained with data augmentation usually generalize better, but the remaining question is to what extent the info Transformation can improve the performance of the CNN model on the image dataset. But with spectrograms, we cannot really rotate, flip, or crop because it will change the info. There we use techniques like time stretching, time masking, frequency masking, and pitch shift.

Octaves	0.5
Bins per Octave	24
Dilation	0.25
Sample Size	0.5
Maximum width	0.2

Table 1: Initial Parameters of the transformer method

Using the hyper-parameters like a number of octaves by which the spectrograms will be shifted for time stretching, a number of bins per octave are defined in data augmentation, and we transform the spectrograms. We shift the spectrogram by 0.5. That is, we take random 5s of data out of 10s to keep the data but take fragments instead of the whole file. Table 1 shows the values we have chosen for the hyperparameters to create the transformer. The values are selected on the default basis, which is suitable but now the best values for the model. Later in, we will find the best suitable hyperparameter values depending upon the best model, which we will save upon training for 200 epochs. The transformed data is then normalized by employing a default normalization vector for pre-trained VGG16. Training and testing transform returned during this method to be employed in the data loader.

With the use of hyperparameters, we train our model for 200 epochs. 1st epoch has an accuracy of 41.2%. Achieving the highest accuracy of 70 % in the 50th epoch. If the accuracy doesn't increase in 10 epochs, we break the loop to save training time and so with an accuracy of 68% in the 68th epoch, we stop training.

3.8 Baseline Model

DCASE 2019 challenge has curated a baseline model for reference based on CNN[30]. For each 10-second signal, log mel-band energies are initially extracted by the system, which then uses two CNN layers and one fully connected layer to train a network to assign scene labels to the audio signals. They have used two acoustic features, and the first one is frame analysis with 50% hop size, and the second is log mel-band energies of 40 bands. The architecture consists of 7 convolutional layers with batch normalization and ReLU activation function. After CNN each layer, a 2D max pooling layer is added. Further, a flattened layer, a dense layer, is added with a dropout layer. Finally, attaching a SoftMax layer. With a batch size of 16, ADAM optimizer and learning rate of 0.001, we train the model on train and validation sets for 200 epochs. Achieving accuracy of 62.5% for the test dataset.

4. Training Procedure

At training time, we show the network normalized data suitable for VGGnet and then create data loaders from the transforms with hyperparameters. To further prevent over-fitting, we also apply pre-emphasis data augmentation [26] with an alpha of 0.2. In this implementation, we use -1 rather than 0 as starting point. As we are working with lower frequencies, we divide the output signal by (1- alpha) to not change the amplitudes further.

As optimizers, we use the ADAM update rule [27] with an initial learning rate of 0.01 and a mini-batch size of 100 samples. Each model is trained for 200 epochs, where we stop learning if the accuracy doesn't increase for the last ten epochs. Figure 5 shows a confusion matrix and accuracy of a single epoch. The columns in the confusion matrix correspond to the predicted categories. The rows to the actual categories. We save the best accuracy model in the checkpoint to use in the evaluation phase. We achieved 70% accuracy in training on the 50th epoch, and the figure beside shows the class names, which clearly depict that the bus has the most accurate predictions while the metro has the least. Metro and metro station are quite a lot of times cross predicted same for airport and shopping mall possibly because of the lower frequencies in both of the scenes.

```
Confusion matrix
[[213   0   1   3   5   9  41  17   1   1]
 [  0 267   2   8   2   0   1   2   3   5]
 [  7  20 147  45   6  12   5   9   5  35]
 [ 30   8   9 154   5  11  43  14   7   9]
 [  3   0   0   1 237  35   0   7   6   1]
 [ 31   1   0   3  11 179  14  35  16   0]
 [ 44   0   0   7   0  12 207  18   2   0]
 [ 38   0   1   5   6  39  20 176   4   0]
 [  2   4   0   4  14  34   0   5 227   0]
 [  4  34  35  30   8   2   1   6   2 168]]
.....................................Epoch: 50/200,  Loss: 0.0188
.....................................Epoch: 50/200,  Loss: 0.0229
.....................................Epoch: 50/200,  Loss: 0.0161
.....................................Epoch: 50/200,  Loss: 0.0203
.....................................Epoch: 50/200,  Loss: 0.0232
.....................................Epoch: 50/200,  Loss: 0.0220
.........................           Epoch 50 completed
----------------------------------------
----------------------------------------
-----------
Accuracy: 70.5274 %
```

```
{'airport': 0,
'bus': 1,
'metro': 2,
'metro_station': 3,
'park': 4,
'public_square': 5,
'shopping_mall': 6,
'street_pedestrian': 7,
'street_traffic': 8,
'tram': 9}
```

Figure 5: Confusion Matrix on train data

4.1 Evaluation

In the evaluation phase, we use the same data processing as in the training phase for the testing process. We have already created a test loader with the same parameters as training to keep the data the same. The best model saved in the training phase is called back but with test loader data. After running the model, we get an accuracy of 70% on the test dataset. Figure 6 shows the confusion matrix of the test data. Like a training matrix, the columns correspond to the predicted categories and the rows to the actual categories. Clearly, buses and parks are the best-recognized scenes. Tram and metro aren't classified so precisely for obvious reasons as they both have very similar sounds, which wasn't recognized so much in the training phase.

```
Accuracy: 69.2892 %
Confusion matrix
[[227   0   4  10   1  14  28   6   0   0]
 [  0 261  10   3   0   0   0   3   0  13]
 [  3  14 165  31   2   1   2   2   0  69]
 [ 26   6  28 167   2  10  19  13   3  16]
 [  1   0   1   3 249  20   0   3   7   6]
 [ 30   1   4   5  15 165  18  33  16   3]
 [ 58   0   1  13   0   7 198  12   0   1]
 [ 54   2   2   7   4  42  20 149   6   3]
 [  0   4   2   5  12  38   1  11 217   0]
 [  2  21  31  16   2   3   0   5   0 210]]
```

Figure 6: Confusion matrix on test data

4.2 Scene Classification

We have created a prediction function where we will classify any random scene from the data using the saved model from the training phase. We use the test transform to normalize the data and then feed it to the model. Further classifying it. We compute the probability of the scene, classifying it as every class. Then plot the probabilities for visualization. Figure 5 shows the results of random audio data from the test loader of a dataset of DCASE 2019.

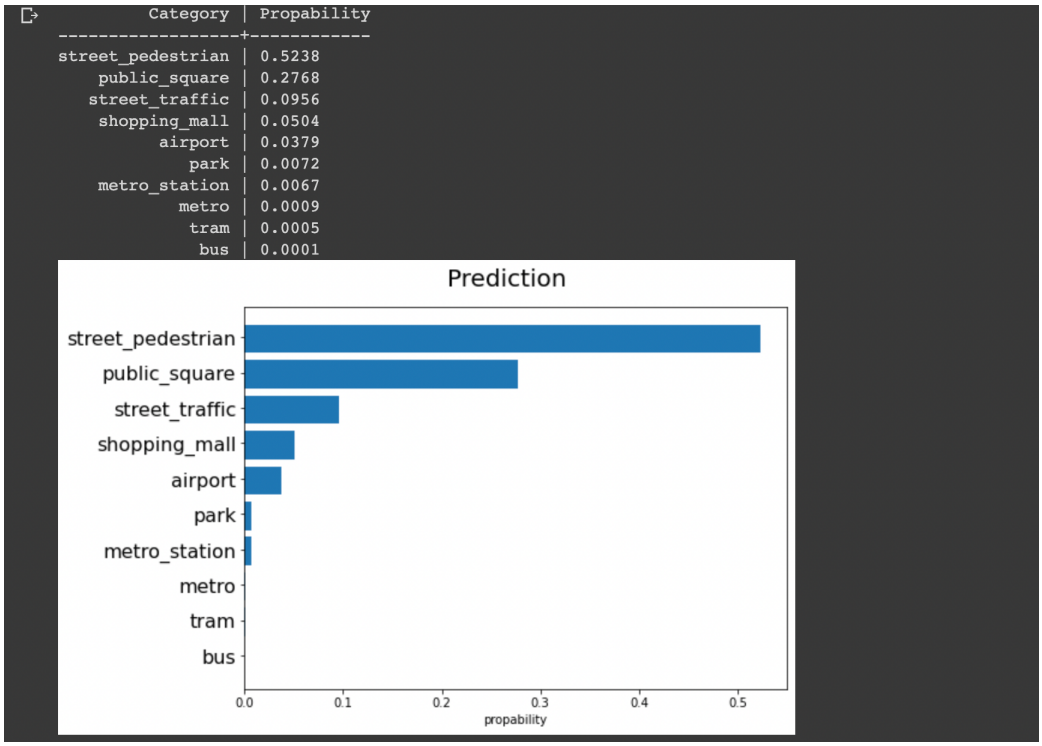


Figure 7: Probabilities of Scene for a Street_Pedestrian Scene

4.3 Model Optimization

Now, we got our best model, acceptable accuracy and predicted some unknown data. We will now find the best values for the hyperparameters for our new model. Here we use the ordinary grid search method to find the best values of hyperparameters and run our model for three different sets of values of each hyperparameter and, similarly, three repetitions with fixed random seeds to reproduce the results.
Initially, we tested model for sample size, maximum width, octaves and dilation with values [0.25 ,0.5,0.75], [0.0,0.1,0.2], [0.25 ,0.5,0.75] and [0.0,0.25,0.5] respectively. We got the best value for sample size as 0.5, which we were already using, for maximum width of 0.0, showing that frequency masking is not helping the model. Octaves and dilation also didn't show any change from the default values we were using.
In the second round, we increased the values a bit more drastically shown in table 2 below, which decreased the accuracy of the model. Clearly, our model works best with initial values. Using more octaves gives the spectrogram a greater shift, and most of the important data is lost. Similarly, increasing pixels isn't also the best-suited method. Dilation causes a change in tempo, which stretches the time, and if we stretch it too much, the lower frequencies are affected, so the classification doesn't work properly.

Parameters	Iteration 1	Iteration 2	Best values
Sample Size	0.25,0.5,0.75	1,2,3	0.5
Maximum Width	0.0,0.1,0.2	0.2,0.3,4	0
Octaves	0.25,0.5,0.75	0.75,1,1.25	0.5
dilation	0.0,0.25,0.5	0.5,0.75,1.0	0.25
Bins per octave	24,48,64	64,128,256	24

Table 2: Hyper Parameters selected for the optimization

5. Results

With the best hyperparameters found, we test the model again with our test data in the optimization method. We used three iterations for every pair. With the first pair, a sample size value of 0.5 and maximum width of 0 was the best value we got. We got 69.8% accuracy with the test data. In the second round where we discovered the best number of octaves is 0.5, and the model works best with dilation of 0.5. With these parameters, we got 72.2% accuracy for the test data. We noticed a slight increase with best parameter values. Also, we discovered that frequency masking wasn't so helpful in our model. At the same time, the fragmentation of audio in half proved to be a great data augmentation technique.

We were also comparing our model with the baseline model and were able to increase the accuracy by almost 10%. Table 3 below shows each scene-wise comparison with the baseline model.

Acoustic Scene	Baseline	Proposed Model
Airport	48.4	66.5
Bus	62.3	87.6
Metro	65.1	69.1
Metro Station	54.5	61.2
Park	83.1	89.9
Public Square	40.7	54.6
Shopping Mall	59.4	66.8
Street Pedestrian	60.9	62.3
Street Traffic	86.7	88.6
Tram	64.0	75.4
Average	62.5	72.2

Table 3: Acoustic Scene-wise accuracy of our model and the baseline model.

6. Conclusions

In this paper, we investigated the transfer learning mechanism of CNN architecture and weight initialization from the image classification to the acoustic scene classification. We experimented on VGGNet-16 CNNs and found that the pre-trained networks from the ImageNet task achieved better performance in acoustic scene classification than the randomly initialized ones. Moreover, by splitting each audio recording into smaller segments added to achieve accurate results, the accuracy could be further improved by using different methods of data augmentation. Validated on the DCASE2019 ASC subtask A, our proposed system achieved an accuracy of 72.2% on the development set. This demonstrates that transfer learning is a better approach when compared to the conventional CNN structure.
The conversion of spectrograms and using the grey scale made it simpler to process the images. The sound quality was improved using the pre-emphasis filter. We split the development data set into train, test, and valid folders to use the deep learning model. Our data augmentation methods did prove to have a positive impact on our model with the optimized parameters. Though frequency masking did not help our model, the random time shifting proved to be a good idea for augmentation.

6.1 Future Work

The experiments were carried out on only the development data set of DCASE 2019 sub-task A. With the evaluation data set, we can get more precise insights into our model. The evaluation data set consists of data from two cities. With a larger dataset, we can develop a fine-tuned version of the evaluation techniques. In the future, we will experiment with Generative Adversarial Networks (GAN) to generate new data from the training phase. With these, adding different data augmentation methods like pitch shifting can help improve accuracy.
Finally, currently, we are working on creating a better classifier for our model to improve the accuracy.

7. ACKNOWLEDGMENT

I want to thank my supervisor Li Zhang for helping me shape, direct, and polish my work for this experiment. Her patience with the process and motivation to explore more about the subject have played a major part in making this project what it is today. I would also like to thank the Royal Holloway University of London for providing me with all the support I needed for the research.

8. REFERENCES

[1] https://github.com/toni-heittola/dcase2019_task1_baseline

[2] https://dcase.community/challenge2019

[3] Mun S, Park S, Han D K, et al. Generative adversarial network based acoustic scene training set augmentation and selection using SVM hyper-plane[J]. Proc. DCASE, 2017: 9397.

[4] Lim H, Park J, Han Y. Rare sound event detection using 1D convolutional recurrent neural networks[C]//Proceedings of the Detection and Classification of Acoustic Scenes and Events 2017 Workshop. 2017: 80-84.

[5] Advance S, Virtanen T. A report on sound event detection with different binaural features[J]. arXiv preprintarXiv:1710.02997, 2017.

[6] Sakashita Y, Aono M. Acoustic scene classification by ensemble of spectrograms based on adaptive temporal divisions[J]. IEEE AASP Challenge on DCASE 2018 technical reports, 2018.

[7] Lukyanov I, Hautamäki V, Lee K A. Recurrent neural network and maximal figure of merit for acoustic event detection[J]. IEEE AASP Challenge on DCASE 2017 technical reports, 2017.

[8] Han Y, Park J, Lee K. Convolutional neural networks with binaural representations and background subtraction for acoustic scene classification[J]. The Detection and Classification of Acoustic Scenes and Events (DCASE), 2017: 1-5.

[9] Dorfer M, Lehner B, Eghbal-Zadeh H, et al. Acoustic scene classification with fully convolutional neural networks and I-vectors[J]. IEEE AASP Challenge on Detection and Classification of Acoustic Scene and Events (DCASE), 2018.

[10] Zeinali H, Burget L, Cernocky J. Convolutional neural networks and x-vector embedding for dcase2018 acoustic scene classification challenge[J]. arXiv preprint arXiv:1810.04273, 2018.

[11] P. Arora and R. Haeb-Umbach, "A study on transfer learning for acoustic event detection in a real-life scenario," 2017 IEEE 19th International Workshop on Multimedia Signal Processing (MMSP), 2017, pp. 1-6, DOI: 10.1109/MMSP.2017.8122258.

[12] Chen, H., Liu, Z., Liu, Z., Zhang, P., & Yan, Y. (2019). Integrating the Data Augmentation Scheme with Various Classifiers for Acoustic Scene Modelling. DCASE2019 Challenge.

[13] https://ai.googleblog.com/2019/04/specaugment-new-data-augmentation.html

[14] https://dcase.community/challenge2019/task-acoustic-scene-classification

[15] Y. Han and K. Lee, “Convolutional neural network with multiple width frequency-delta data augmentation for acoustic scene classification,” IEEE AASP Challenge on Detection and Classification of Acoustic Scenes and Events, 2016.

[16] R. Hyder, S. Ghaffarzadegan, Z. Feng, and T. Hasan, “But bosch consortium (b2c) acoustic scene classification systems for case 2017 challenge.”
[17] J. Xu, Y. Zhao, J. Jiang, Y. Dou, Z. Liu, and K. Chen, “Fusion model based on convolutional neural networks with two features for acoustic scene classification.”

[18] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein et al., “Imagenet large scale visual recognition challenge,” International Journal of Computer Vision, vol. 115, no. 3, pp. 211–252, 2015.

[19] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” arXiv preprint arXiv:1409.1556, 2014.

[20] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.

[21] A. Satt, S. Rozenberg, and R. Hooray, “Efficient emotion recognition from speech using deep learning on spectrograms,” Proc. Interspeech 2017, pp. 1089–1093, 2017.

[22] Kensett A, Harrison PJ, Spjuth O. Transfer Learning with Deep Convolutional Neural Networks for Classifying Cellular Morphological Changes. SLAS DISCOVERY: Advancing the Science of Drug Discovery. 2019;24(4):466-475. doi:10.1177/2472555218818756

[23] https://neurohive.io/en/popular-networks/vgg16/

[24] https://medium.com/analytics-vidhya/understanding-the-mel-spectrogram-fca2afa2ce53

[25] https://pytorch.org/docs/stable/generated/torch.nn.LogSoftmax.html

[26] https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html

[27] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv preprint arXiv:1412.6980, 2014.

[28] https://towardsdatascience.com/adam-optimization-algorithm-1cdc9b12724a

[29] https://towardsdatascience.com/cross-entropy-negative-log-likelihood-and-all-that-jazz-47a95bd2e81

[30] https://github.com/toni-heittola/dcase2019_task1_baseline/blob/master/task1a.py

[31] https://doi.org/10.5281/zenodo.2589280

[32] https://github.com/xypron/acoustic_sceneclassification

9. APPENDIX

MyProject folder consists of ASR.ipynb and Report. To run the code please download data from the link in reference 31. Combine all the audio files from the different zip folders to one folder suggested to be named as "Audio". After loading the audios Convert method will convert all the audios into spectrograms and create a folder named spectrograms in the same file. The spectrograms file will be split in train valid and test dataset. These will be used to process further.